future internet

MDPI

Article

A Modified TCP BBR to Enable High Fairness in High-Speed Wireless Networks

Jinlin Xu [1,2], Wansu Pan [1], Haibo Tan [1,2,*], Longle Cheng [1], Xiru Li [1] and Xiaofeng Li [1,2]

[1] Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei 230031, China; jlxu@hfcas.ac.cn (J.X.); wansupan@mail.ustc.edu.cn (W.P.); llcheng@hfcas.ac.cn (L.C.); xrli@hfcas.ac.cn (X.L.); xfli@hfcas.ac.cn (X.L.)

[2] Science Island Branch, Graduate School of USTC, Hefei 230026, China

[*] Correspondence: hbtan@hfcas.ac.cn

Abstract: Wireless networks, especially 5G and WiFi networks, have made great strides in increasing network bandwidth and coverage over the past decades. However, the mobility and channel conditions inherent to wireless networks have the potential to impair the performance of traditional Transmission Control Protocol (TCP) congestion control algorithms (CCAs). Google proposed a novel TCP CCA based on Bottleneck Bandwidth and Round-Trip propagation time (BBR), which is capable of achieving high transmission rates and low latency through the estimation of the available bottleneck capacity. Nevertheless, some studies have revealed that BBR exhibits deficiencies in fairness among flows with disparate Round-Trip Times (RTTs) and also displays inter-protocol unfairness. In high-speed wireless networks, ensuring fairness is of paramount importance to guarantee equitable bandwidth allocation among diverse traffic types and to enhance overall network utilization. To address this issue, this paper proposes a BBR–Pacing Gain (BBR–PG) algorithm. By deriving the pacing rate control model, the impact of pacing gain on BBR fairness is revealed. Adjusting the pacing gain according to the RTT can improve BBR's performance. Simulations and real network experiments have shown that the BBR–PG algorithm retains the throughput advantages of the original BBR algorithm while significantly enhancing fairness. In our simulation experiments, RTT fairness and intra-protocol fairness were improved by 50% and 46%, respectively.

Keywords: congestion control; BBR; pacing gain; high fairness; low retransmission

check for updates

Citation: Xu, J.; Pan, W.; Tan, H.; Cheng, L.; Li, X.; Li, X. A Modified TCP BBR to Enable High Fairness in High-Speed Wireless Networks. *Future Internet* **2024**, *16*, 392. https://doi.org/10.3390/fi16110392

Academic Editor: Jose I. Moreno Novella

Received: 3 October 2024
Revised: 21 October 2024
Accepted: 21 October 2024
Published: 25 October 2024

1. Introduction

In 2016, Google [1] proposed a new congestion control algorithm (CCA) based on Bottleneck Bandwidth and Round-Trip propagation time (BBR). BBR creates a network path model by measuring the available bottleneck bandwidth (BtlBw) and the Round-Trip propagation time (RTprop) to maximize delivery rate and minimize latency. Its improved performance and adaptability have led to widespread acclaim and interest since its launch. In high-speed wireless networks such as 5G or WiFi, there are a variety of applications that include both bandwidth-demanding long flows (e.g., video surveillance drone applications and Augmented Reality (AR)/Virtual Reality (VR)) and latency-sensitive short flows [2,3]. In 5G network environments, the role of high-bandwidth long Round-Trip Time (RTT) flows and frequent short RTT flows are more pronounced, and they have different Quality of Service (QoS) and RTT requirements.

Unfortunately, BBR cannot fairly share bandwidth with traffic with different RTTs. The first version of BBR (BBRv1) overestimated the sending rate by constantly filling the Bandwidth Delay Product (BDP), leading to problems such as queue stacking and RTT unfairness [4–6]. To overcome these limitations, BBRv2 (BBR version 2) [7] introduced mechanisms to better estimate and adapt to the actual available bandwidth and to minimize queue stacking. Although BBRv2 adapts better to network changes than BBRv1, RTT unfairness remains when competing with other flows. Google has recently released BBRv3

(BBR version 3) [8], and while BBRv3 exhibits greater fairness to loss-based CCAs, it falls short in Next-Generation Networks (NGNs) with widely varying RTT requirements.

In this paper, we mitigate the observed fairness challenges of BBR in high-speed but lossy networks, such as WiFi and 5G networks. BBR has been deployed in some application scenarios, but more research is needed to further improve BBR to ensure that there are no more potential vulnerabilities. As a result of this motivation, this paper makes the following contributions:

1. The pacing rate control model is established, and the origin of BBR fairness issues is revealed. Because the RTprop and queuing delay affect the original pacing gain, the RTT fairness and inter-protocol fairness of BBR result.
2. The optimization algorithm BBR–Pacing Gain (BBR–PG), which adaptively adjusts the original fixed pacing gain to balance the sending rates of different flows in the bottleneck queue, is proposed. The phase time of the original BBR is optimized to accelerate the flow convergence.
3. BBR–PG is compared and analyzed by simulation and using a real network. The results show that BBR–PG maintains the throughput advantage of the original BBR, improves RTT fairness and inter-protocol fairness, and reduces retransmission.

The rest of this article is arranged as follows: Section 2 introduces the principle of BBR and related optimization research. The theoretical model and derivation of the proposed algorithm are in Section 3. Section 4 is experimental results and analysis. Conclusions are discussed in Section 5.

2. Related Work

Since the launch of BBR, several studies have identified certain problems in BBRv1 (BBR version 1). Hock et al. [4] observed that there is a bandwidth discrepancy between elephant and mouse flows sharing the bottleneck link due to the higher BDP of the elephant flow. Scholz et al. [5] found that long queues are created during the startup phase, inhibiting existing traffic. Scherrer et al. [6] found that in cases where there are multiple flows managed by BBRv1, overestimated transmission rates can lead to standing queues exceeding 1.5 times the BDP, resulting in packet loss and unfair bandwidth sharing between flows. To address these issues, Google introduced BBRv2 [7] in 2018, adding Explicit Congestion Notification (ECN) and packet loss rates. In their evaluation of various BBR versions, BBRv2 could not provide fair bandwidth sharing in the presence of other BBR flows with different RTTs and loss-based TCP flows [9,10]. Nandagiri et al. [11] conducted an experimental evaluation comparing BBRv1 and BBRv2 and found that BBRv2 still consumes more bandwidth for long RTT flows in networks with larger buffers. Drucker et al. [12] pointed out BBRv2's improvements in terms of traffic fairness and improved coexistence with CUBIC and Reno. However, they also noted that BBRv2 trades performance for better fairness under losses. The BBRv3 [8] aims to fix the bugs in BBRv2 and optimize the performance parameters. Despite the updates and improvements in BBRv2 and BBRv3, experiments show that RTT unfairness still exists [13].

With regard to inter-protocol fairness, Hurtig et al. [14] evaluated BBR's inter-protocol fairness when the traffic consists of flows of different sizes. Zhang et al. [15] proposed Modest BBR, which adjusts the rate according to the network situation, reduces retransmission, and realizes high throughput and inter-protocol fairness. Ware et al. [16] provided the first model capturing BBR's behavior in competition with loss-based CCAs. Song et al. [17] proposed BBR convergence window scaling (BBR–CWS), which adjusts the size of CWND through packet loss feedback. It reduces excessive retransmission and finds a balance between algorithms. Ishikura et al. [18] proposed a BBR with Adjusting RTprop (BAR), which adjusts RTprop so that BBR obtains fair bandwidth with CUBIC.

To solve the RTT fairness problem, Ma et al. [19] proposed the BBQ algorithm. By setting the upper limit of the detection time, long RTT flows are prevented from preempting the bandwidth of short RTT flows. When no persistent queue is detected, BBQ reverts to the original BBR mechanism to maintain bandwidth detection speed. Yang et al. [20] proposed an Adaptive BBR algorithm, which alleviates the fairness between different RTT flows by changing the detection period and pacing gain of BBR. Kim et al. [21] proposed Delay–Aware BBR, which improves the fairness between different RTT flows. In our previous research, we also optimized RTT fairness by adjusting the sending rate [22] to alleviate RTT unfairness. Njogu et.al. [23] introduced BBR–With Enhanced Fairness (BBR–EFRA), dynamically adjusting Congestion Window (CWND) based on buffer queue status for equitable competition among different RTT flows. However, these algorithms rely on accurate real-time queue estimation, which is challenging in dynamic network conditions.

BBRs have been applied to different network scenarios to improve service quality [24–26]. Guo et.al. [27] have developed an enhanced Stateful–TCP technique that converts BBRs into new S–BBRs to accelerate their startup performance and improve throughput in 5G networks. For BBRs in Wi-Fi and 5G networks, Xie et al. [28] proposed an improved BBR, called Yinker, which dynamically adjusts the pacing gain of BBR based on network conditions, including loss rate and congestion level. Ahsan et.al. [29] proposed BBR–n (BBR new), which provides a better throughput than the generic BBR v2 in WiFi networks.

When BBR is applied, some users may exploit the fairness loophole of BBR to compete for bandwidth maliciously. Therefore, to provide potential improvements to the final version of BBR, it is important to continue to investigate ways to improve RTT fairness and inter-protocol fairness.

3. Proposed Work

3.1. Overview of BBR

In this section, the influences of pacing gain and convergence period on BBR performance are studied by deriving the BBR pacing rate control model. Figure 1 shows the state machine and network path model in BBR. It can be observed that BBR mainly controls the transmission of data packets through two aspects: pacing rate and CWND. Therefore, to solve the complex problem of BBR, the pacing rate control model is established and derived to find the near-optimal solution.

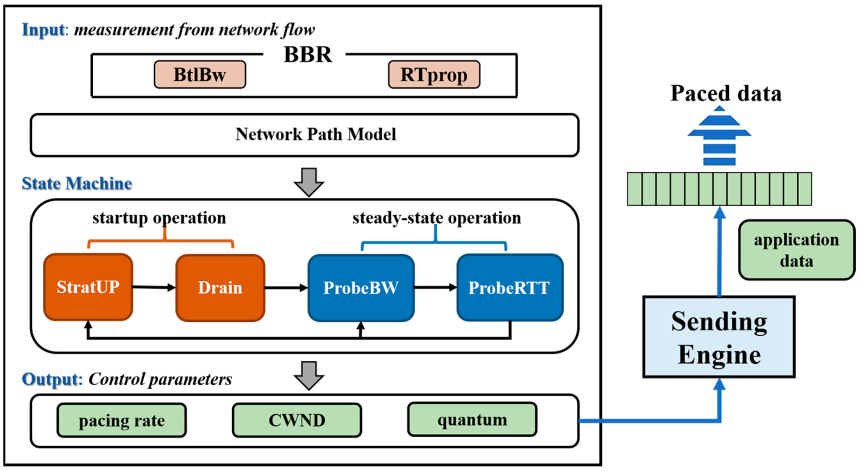


Figure 1. The state machine and network path model.

3.2. BBR Fluid Model

By establishing a fluid model, we simplify the operational mechanism of BBR. We assume that n flows with different RTTs passing through a bottleneck link with a bandwidth of C. Let $flow_i$ ($i \in$ [1; n]) represent $flow_i$ and d_i(t) represent the delivery rate at time t. In the ideal state, $d_1 + d_2 + \ldots + d_n$ = C. Let T_i(t) represents the RTT of $flow_i$ at time t, which is calculated as follows:

$$T_i(t) = T_{D_i}(t) + T_{p_i} = \frac{q_i(t)}{C} + T_{p_i} \tag{1}$$

where T_D denotes queuing delay, q_i represents queue length, and T_p represents RTprop.

Let I_i represent the inflight data of $flow_i$, namely, the upper bound of the bottleneck link, which is calculated as follows:

$$I_i(t) = d_i(t) \times T_i(t) = d_i(t) \times (\frac{q_i(t)}{C} + T_{p_i}) \tag{2}$$

The estimated BtlBw of the $flow_i$ at time t can be obtained with

$$BtlBw_i(\mathrm{t}) = max(d_i(\mathrm{t}))(T \in [t - 10RTT;\ t]) \tag{3}$$

Combined with Equation (2), $flow_i$ can obtain a 1.25-fold gain in the first probe-up cycle, so the maximum delivery rate at time t is shown as

$$\max(d_i(t)) = \frac{\max(I_i(t))}{T_i} = \frac{1.25 \times T_{p_i} \times BtlBw_i(t - \Delta t)}{T_i} \tag{4}$$

The detection cycle of BBR flow is 8RTprop, so the estimated BtlBw of $flow_i$ in the new round is updated as

$$BtlBw_i(t) = \frac{1.25 \times T_{p_i} \times BtlBw_i(t - 8T_{p_i})}{T_i} \tag{5}$$

Assuming that the total bandwidth is 1, the relationship between pacing gain and bandwidth is shown in Figure 2, where A and B represent two different flows, respectively. The total bandwidth and the initial bandwidths of the A flow and the B flow are assumed to be 1, a, and $1 - a$, respectively. As show in Figure 2, if A and B are, respectively, probe-up once, the bandwidth increment expressions of A and B can be obtained as follows:

$$\Delta A = \frac{1-a}{0.25a+1} - \mathrm{a} = \frac{0.25a\ (1-a)}{0.25a+1} \tag{6}$$

$$\Delta B = \frac{1.25(1-a)}{1.25-0.25a} - (1-a)\ = \frac{0.25a(1-a)}{1.25-0.25a} \tag{7}$$

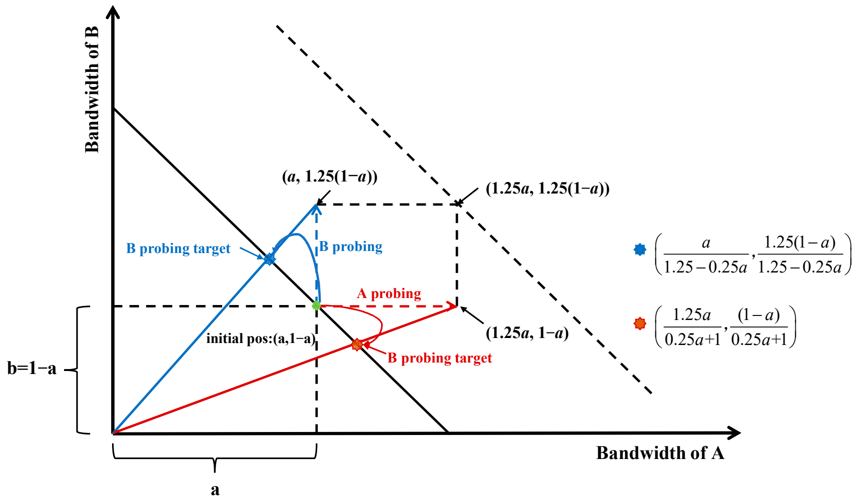


Figure 2. The relationship between pacing gain and bandwidth.

It can be seen from Equations (6) and (7) that the two flows intersect at a = 0.5 and are symmetric at about a = 0.5. The RTT of different flows on the same bottleneck link will not be the same, and with its initial bandwidth, it is difficult to achieve fair sharing.

The influence of RTT difference on bandwidth occupancy was further deduced. The RTTs of $flow_1$ and $flow_2$ are set to T_1 and T_2, respectively, and $T_2 = \lambda T_1$ ($\lambda \geq 1$). The total inflight data in the time interval [t, $t + \lambda$] of the two flows can be calculated by Equations (8) and (9), where $\alpha = \lfloor (t-1) \times \lambda \rfloor$.

$$I_1(t) = [d_1(\alpha) \times \lambda t + d_1(\alpha - 1) \times (1 - \lambda t)] \times T_1 \tag{8}$$

$$I_2(t) = d_2(t-1) \times T_2 \tag{9}$$

The bandwidth occupation oBw of different flows can be determined by I_i:

$$oBw_i(t) = C \times \frac{I_i(\mathrm{t})}{\sum_{i=0}^{n} I_i(t)} \tag{10}$$

In substituting Equations (8) and (9) into Equation (10),

$$oBw_2(t) = C \times \frac{I_2(t)}{I_2(\mathrm{t}) + I_1(t)} = C \times \frac{\lambda \times d_2(t-1)}{\lambda \times d_2(t-1) + I_1(t)} \tag{11}$$

From Equation (11), the bandwidth occupation of $flow_2$ is related to ratio λ. When d_i increases, oBw_2 also increases, which means that $flow_2$ preempts the bandwidth of $flow_1$. As a result, the BBR's throughput is affected by the RTT ratio.

From the above derivation, it can be seen that the long RTT flow has a larger pacing gain in the ProbeBW phase, resulting in a higher sending rate. If BBR misjudgment or other flows result in a queuing delay where T_D is greater than $0.25T_p$ but less than T_p, the pacing gain will become negative, and the sending rate will decrease instead of increasing. This is also the reason for bandwidth unfairness when BBR and other CCAs are deployed together. In addition, the stable convergence of BBR can be realized only when all flows stop increasing their sending rate. Therefore, the pacing gain is adaptively adjusted through RTT change to balance the sending rates of different flows.

3.3. The Optimization Algorithm: BBR-PG

In the pacing-controlled phase, the actual pacing gain is as follows:

$$pg = \frac{1.25T_{p_i}}{T_i} = \frac{1.25T_{p_i}}{T_{D_i} + T_{p_i}} \tag{12}$$

The impact of the relationship between queuing delay and RTprop on the bandwidth can be seen from Equation (12). The BBR bandwidth detection process generally does not reach 1.25 times pg unless the whole link is not queued at all. If the queuing time increases significantly, the pg will increase significantly with the increase in T_p. This is the root cause of the RTT unfairness of BBR.

If γ is the pg regulatory factor, the bandwidths of the two flows with different RTTs are the same in the ideal fair case, and the ratio of BDP is the ratio of RTT,

$$\frac{\gamma_1 \times T_{p1} \times BtlBw_1(t)}{\gamma_2 \times T_{p2} \times BtlBw_2(t)} = \frac{T_1}{T_2} = \frac{T_{p1} + \mathrm{T}_D}{T_{p2} + \mathrm{T}_D} \tag{13}$$

If $BtlBw_1$ = BtlBw_2, then:

$$\frac{\gamma_1 \times T_{p_1}}{\gamma_2 \times T_{p_2}} = \frac{T_{p_1} + \mathrm{T}_D}{T_{p_2} + \mathrm{T}_D} \tag{14}$$

The general term of γ is

$$\gamma = g(T_p) = \frac{T_p + T_D}{T_p} = 1 + \frac{T_D}{T_p} \tag{15}$$

The regulatory factor γ is a subtractive function greater than 1, with T_p as the independent variable. The queuing delay T_D is equal to the difference between the actual RTT and RTprop, so γ can be obtained as follows:

$$\gamma = \frac{T_p + sRTT - T_p}{T_p} = \frac{sRTT}{T_p} \tag{16}$$

where *sRTT* represents the actual measured RTT, and its value is the average result of the moving index of RTT.

After the adjusting of the γ, have the RTT fair bandwidth iterative equation:

$$BtlBw(t) = \frac{\gamma T_p}{T_p + T_D} \times BtlBw(t - 8T_p) = BtlBw(t - 8T_p) \tag{17}$$

It can be seen from Equation (17) that the measurement bandwidth *BtlBw* does not increase and reaches equilibrium. In the γ expression g(T_p), the larger the T_p, the smaller the *pg*. Under this guarantee, different RTT flows can equal probe-up bandwidth.

In addition, the window of max-filtered bandwidth needs to be adjusted to match the fixed interval of a steady cycle, so it is adjusted to

$$WIN = \frac{T_{interval}}{T} + 4 \tag{18}$$

where *WIN* is the window of max-filtered bandwidth. $T_{interval}$ is set to 180 ms in this paper, roughly equivalent to the 6 rounds of 30 ms flows.

According to Equation (18), it can be guaranteed that at least the max-filtered bandwidth window can cover the interval between two probe-up cycles and thus can be detected in the max bandwidth window when probe-up. BBR-PG was implemented by modifying the *get_pacing_gains* () and *bbr_update_bw* () functions in the BBR source code. The pseudocode of the ProbeBW phase in BBR-PG is shown in Algorithm 1.

Algorithm 1: ProbeBW phase in BBR-PG

```
Input: rtt_us/*RTT*/, min_rtt_us /*Minimum RTT */, pg
Output: pacing_gain
Initialization: $T_{interval}$ = 180 ms, rtt_cnt = 0
bbr_update_min_rtt() /*Track min RTT seen in the min_rtt filter window*/
if bbr->mode == ProbeBW then
    get_pacing_gains ()
    if bbr->cycle_idx == 0 then
        $\gamma = \frac{rtt\_us}{min\_rtt\_us}$
        pacing_gain = pacing_gain × γ /*Update pacing_gain*/
        return pg
        bbr_update_bw () /*Update BtlBw */
        rtt_cnt++;
        if BtlBw >= bbr_max_bw then
            $WIN = \frac{T_{interval}}{T} + 4$;
            minmax_running_max (WIN, rtt_cnt) /*Incorporate new sample into BtlBw filter.*/
        end if
end if
```

Next, the inter-protocol fairness optimization when BBR and CUBIC coexist is discussed. In addition to the above *pg* interference, the cycle of the ProbeRTT phase also affects inter-protocol fairness. BBR enters the ProbeRTT phase at a fixed interval of 10s to remeasure RTprop. Due to the existence of CUBIC, the measurement of RTprop will be inaccurate. When the BBR flows enter the ProbeRTT phase, the active detection of bandwidth by the CUBIC flows will fill the queue. Because the queue is not empty at this time, the measured RTprop of the BBR flows is larger than the actual RTprop. A larger RTprop will result in a larger CWND, resulting in more inflight, which will benefit BBR flows measured with a lower RTprop than before. This process is repeated throughout the data transmission time, and continuous oscillations can lead to unfairness between protocols, regardless of whether the two flows achieve a fair average throughput.

This problem is solved by randomizing the time to enter the ProbeRTT phase. The fixed interval of 10 s in the original BBR mechanism is changed to a random interval with a mean of 10 s the next time to enter the ProbeRTT phase. The time interval adopted in this paper is calculated as follows:

$$T_{rand} = rand()\%(h_2 - h_1 + 1) + h_1 \tag{19}$$

where T_{rand} is the random interval time and h_1 and h_2 are set to 5 and 15, respectively, which means the value range is set from 5 s to 15 s.

By employing random intervals, different BBR flows no longer enter the ProbeRTT phase at the same time. Some BBR flows enter the ProbeRTT phase and measure RTprop through the empty queue. Other BBR/CUBIC flows that are not in the ProbeRTT phase can actively compete for the free bandwidth in 200 ms. In this way, BBR–PG sacrifices a small amount of bandwidth to alleviate aggression and retransmission, thus achieving better fairness.

4. Evaluation of Experiments

4.1. Testbed Setup

This section uses Network Simulator 3 (NS3), 5G, and WiFi networks to evaluate the performance of BBR–PG. The experimental topology is shown in Figure 3. NS3 is a network emulator that creates a virtual network running multiple hosts, links, and switches on a single machine. Based on the BBR implementation framework, a large number of simulation experiments have been carried out on NS3, where the condition variables include bottleneck bandwidth, RTT buffer size, etc. To eliminate uncontrollable network disturbances (e.g., background traffic, special token bucket policy procedures, etc.), the channel section was simulated using Netem/TBF and configured with different network environments. We set the default active queue management to the Drop-Tail policy and each packet size to 1KB.

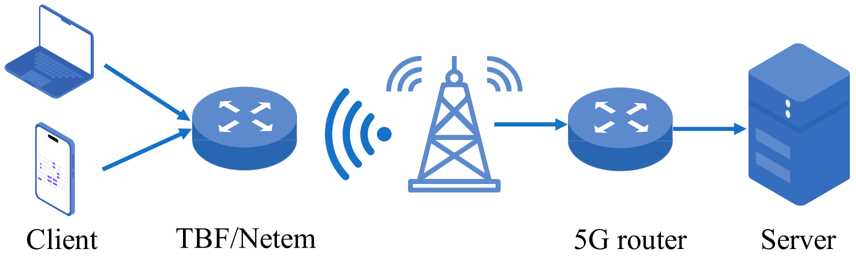


Figure 3. Experimental topology of the network.

The wireless network consists of four physical Linux hosts in the lab and two cloud servers (Node 1 in Beijing, China, and Node 2 in Silicon Valley, CA, USA). The servers have a latency of approximately 50 ms and 230 ms, respectively. For the 5G experiments, we used TCP CCAs implemented in the Linux kernel at the server, and the 5G mobile terminals were used as receivers to measure TCP and application performance over 5G networks.

Moreover, to quantify the fairness at different buffer sizes, the Jain fairness index is introduced [30]. The Jain fairness index can be used to measure fairness in resource competition. The calculation method uses Equation (20), where n represents n sender and x_i is the throughput of the i link.

$$J = \frac{(\sum_{i=1}^{n} x_i)^2}{n\sum_{i=1}^{n} x_i^2} \tag{20}$$

The Jain fairness index (hereafter referred to as the fairness index) can reflect the difference in throughput well, with values ranging from [0, 1]. The closer the fairness index is to 1, the better the fairness of bandwidth allocation is.

4.2. Throughput

We first tested the throughput of the CCAs in a WiFi network environment. The Web service was set up on the server via Nginx, and 10 Gbps files were prepared for download. The CCAs were tested in the same period in each experiment. The statistical results are shown in Figure 4. Except for CUBIC, the download speeds of the algorithms are the same, which improves the transfer speed. CUBIC has the slowest speed. In the experimental process, we found that the speed of BBQ would increase rapidly after the beginning, and then, the speed would decrease significantly, which could be seen in many experiments. On the other hand, BBR and BBR–PG download speeds are relatively stable. In the beginning, the download speed rose rapidly, then fell back slightly, and finally remained stable. In Figure 4b, the speeds of several algorithms increase slightly compared to the small buffer results shown in Figure 4a. Overall, BBR and its variants are effective in improving throughput by about 40% over CUBIC.

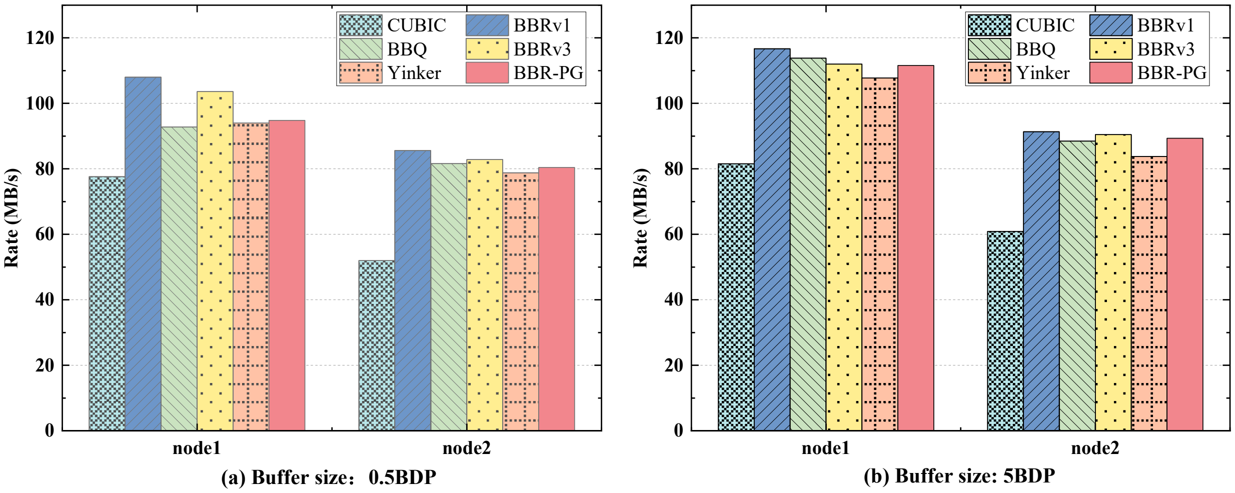
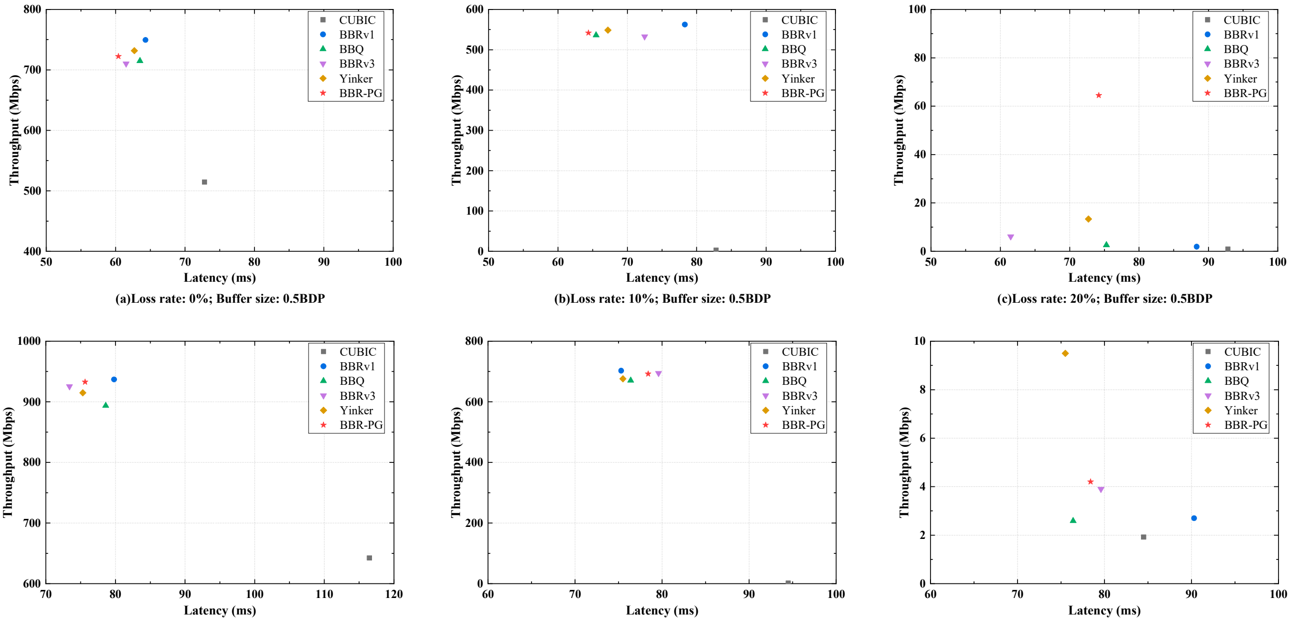


Figure 4. Transmission speed in WiFi networks.

Then, we compare the performance of the CCAs in our extensive simulations for 5G. Figure 5 shows the average throughput and latency of each of the algorithms at different loss rates and buffer sizes. As the loss rate rises, the throughputs of all algorithms decline. BBR and its variants are only capable of maintaining approximately 75% bandwidth utilization when the packet loss rate is 10%. In particular, the throughputs of BBQ and BBRv3 are significantly reduced due to the inability of these algorithms to detect the bandwidth with an appropriate pg. Yinker's throughput is maintained due to its utilization of the loss rate and congestion degree to regulate the pg, which in turn can resist packet loss. In contrast to Yinker, BBR–PG attains a high throughput by modifying the pg in the ProbeBW cycle from a fixed interval to adaptive regulation, thereby counteracting the throughput degradation observed in the lossy network. This further suggests that a fixed pg cannot adapt to the dynamic characteristics of a cellular network, in contrast to an adaptive pg.



(a)Loss rate: 0%; Buffer size: 0.5BDP

(b)Loss rate: 10%; Buffer size: 0.5BDP

(c)Loss rate: 20%; Buffer size: 0.5BDP

(d)Loss rate: 0%; Buffer size: 5BDP

(e)Loss rate: 10%; Buffer size: 5BDP

(f)Loss rate: 20%; Buffer size: 5BDP

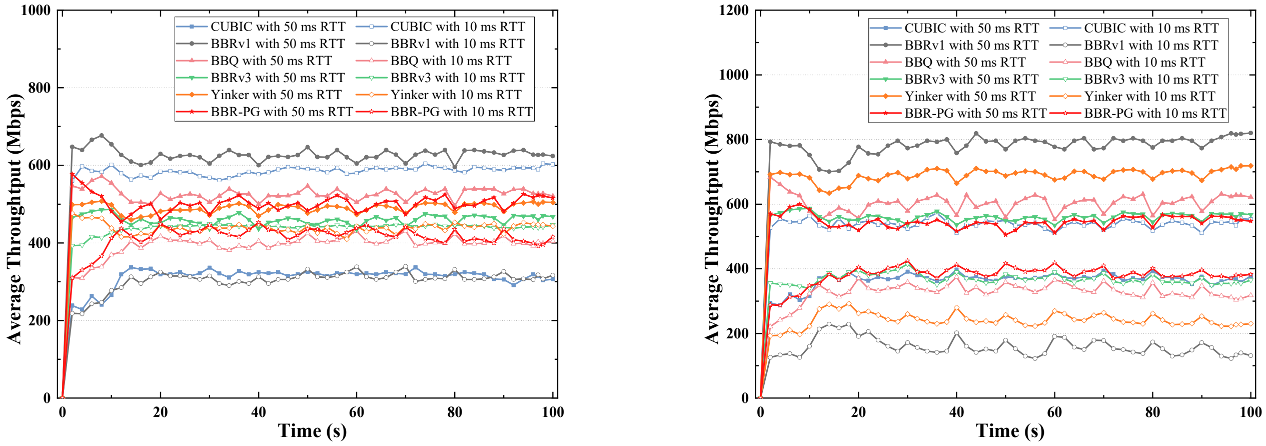
Figure 5. Throughput and latency in 5G networks.

BBR–PG exhibits the lowest latency performance among all CCAs. This is because, as it employs RTT to regulate the *pg*, it can effectively limit the queue length. Consequently, it attains a lower latency when the channel is stable. CUBIC maintains a continuous filling of the buffer until a packet loss occurs. Upon the occurrence of a packet loss, the throughput of CUBIC diminishes. Even in the absence of packet loss, CUBIC's throughput performance is considerably inferior to that of other products. This is because CUBIC is designed for wired networks and cannot adapt to highly dynamic networks such as 5G.

4.3. RTT Fairness

First, we tested the RTT fairness of the algorithm under different conditions using NS3. The experiments were set up with 0.5 BDP and 5 BDP buffers, with 10 ms RTT traffic competing with 50 ms RTT traffic for a 1 Gbps bottleneck bandwidth. The throughput comparison results are shown in Figure 6. For BBRv1 and its variants, the throughput difference between flows increases significantly with buffer size. The throughput difference between the two flows in BBRv2 is smaller than that of BBRv1 and is even smaller than that of CUBIC. While the BBR–PG algorithm has a larger throughput difference in small buffers than BBRv3, the throughput difference in large buffers does not increase further and is significantly smaller than that of BBRv3. Overall, the RTT fairness of BBR–PG is significantly improved compared to BBRv1.

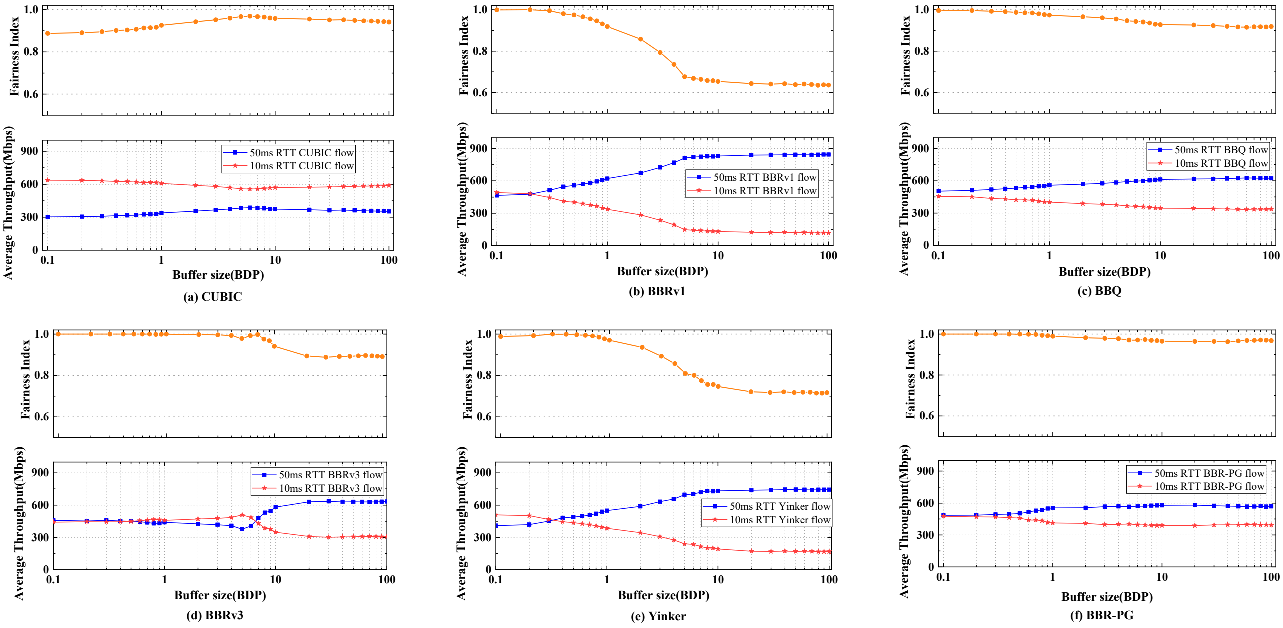
Next, the fairness of the CCAs at different buffer sizes is compared. In the 0.1 BDP~100 BDP buffer, the average throughput and fairness index of 10 ms RTT flows and 50 ms RTT flows are provided in Figure 7. The throughput difference becomes larger with the increase in buffer size. In Figure 7a, the RTT fairness of CUBIC deteriorates with the increase in the RTT ratio, and the final fairness index is about 0.89. Comparing Figures 7b and 8d, it can be seen that the fairness of BBRv3 is improved compared to BBRv1, and as the buffer size increases, the throughput of 10ms RTT flows in BBRv3 is inverted from the throughput of 50 ms RTT flows. In general, BBR–PG has better RTT fairness than the other three algorithms, especially in the larger buffer. Compared with BBR, the fairness index increased by about 53%.



(a) Buffer size: 0.5BDP

(b) Buffer size: 5BDP

Figure 6. Average throughput comparison of 10 ms RTT flows and 50 ms RTT flows.



(a) CUBIC

(b) BBRv1

(c) BBQ

(d) BBRv3

(e) Yinker

(f) BBR-PG

Figure 7. Average throughput and fairness index of 10 ms RTT flows competing with 50 ms RTT flows in different buffer sizes.

In addition to buffer size having a significant impact on RTT fairness, RTT differences also significantly affect RTT fairness. As depicted in Figure 8, with the increase in RTT difference, the throughput of the long RTT flow gradually dominates, and the fairness index gradually decreases. The RTT fairness of BBRv3 is improved over BBRv1, which has a fairness index of only 0.59 compared to 0.89 for BBRv3. In Figure 8f, BBR–PG can maintain good RTT fairness, and the fairness index can be maintained around 0.95. On the whole, BBR–PG can maintain high fairness at different RTT ratios. Especially when the RTT ratio is greater than 4, the fairness index of BBR–PG, which improves by 61% compared with BBRv1, is the highest.

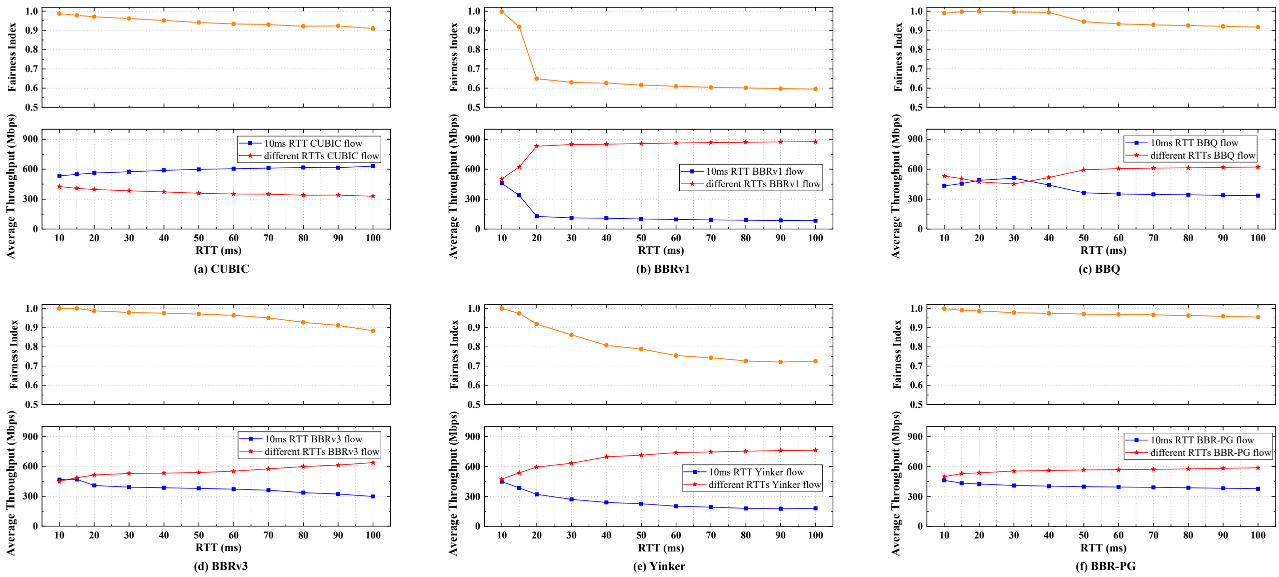


Figure 8. Average throughput and fairness index of 10 ms RTT flows coexisting with different RTT flows in 5BDP buffer.

Then, experiments were carried out in a 5G communication environment. The packets were transmitted (downlink communication) from the base station to the mobile terminal. Figure 9 shows the statistical results of the throughput ratio of 10 ms and 50 ms RTT flows and the corresponding fairness index. As observed from the results provided in Figure 9, 50 ms RTT flows have obvious advantages in different buffer sizes. Although the disadvantage of 10 ms RTT flows is alleviated in shallow buffers, their bandwidth share is still far below its reasonable share. Unlike the other five CCAs, CUBIC's 10 ms RTT flows are more advantageous in shallow buffers. BBRv3 and Yinker have improved fairness compared to BBRv1. Specifically, in the 0.5 BDP buffer, the fairness index of BBR–PG is close to 0.992, with an 11% improvement in RTT fairness. In the 5BDP buffer, the fairness index was approximately 0.966, with a 9% improvement in RTT fairness. This further proves the effectiveness of the BBR–PG algorithm in the simulation results.

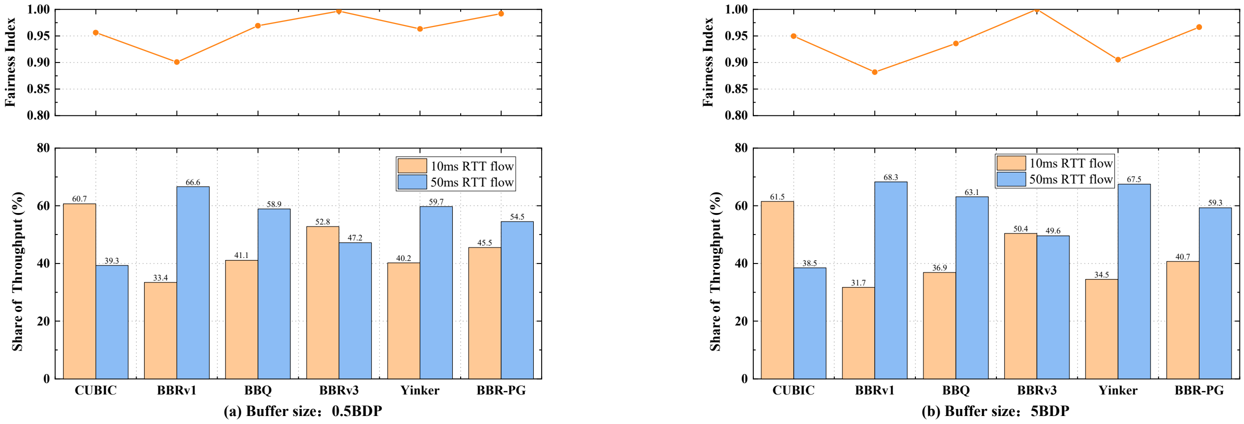


Figure 9. Throughput shares and fairness index in 5G network.

Meanwhile, an RTT fairness test was carried out in the WiFi network to verify the performance of each CCA. We deployed locally installed centralized CCAs and chose the Beijing node as the cloud server. As we can see from Figure 10, the experiment results of the WiFi network are similar to those of NS3 and 5G. The share of throughput between the 50 ms RTT flow and 10ms RTT flow in the WiFi network also has a certain deviation.

The deviation in bandwidth sharing between the two flows in the WiFi network is further widened compared to the NS3 and 5G network experiments. Especially in Figure 10b, the fairness index of BBRv1 is only 0.824, which is significantly lower than that of several other CCAs. In comparison, BBR-PG's fairness is improved by 17% compared to BBRv1.

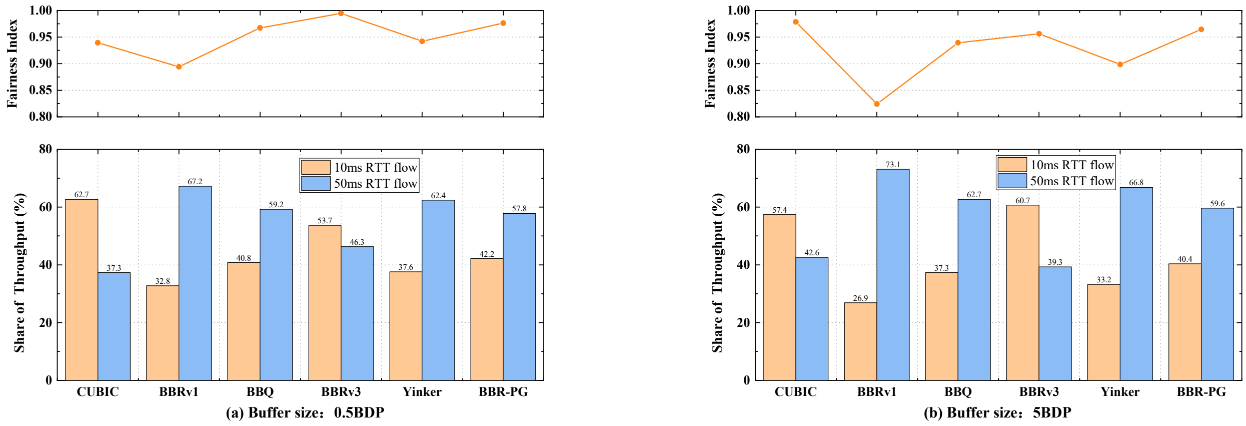


Figure 10. Throughput shares and fairness index in WiFi network.

4.4. Inter-Protocol Fairness

We conduct inter-protocol fairness experiments based on WiFi networks. A comparison of the throughput shares of the CCAs when BBR and its variants coexist with CUBIC is shown in Figure 11. In Figure 11a, BBR and its variants have a bandwidth-robbing advantage at different bottleneck bandwidths in small buffer sizes, and CUBIC is always at a disadvantage. In Figure 11b, CUBIC shows a throughput advantage when the bottleneck bandwidth is small, but as the bottleneck bandwidth increases, BBR and its variants can achieve a higher throughput. This is because CUBIC, being a loss-based CCA, tends to fill the bottleneck buffer until it runs out, regardless of the size of the buffer. However, the comparison between the BBR variants shows that the inter-protocol fairness of BBR–PG is much better than that of BBR for different bandwidth conditions and buffer sizes, suggesting that BBR–PG can improve intra-protocol fairness.

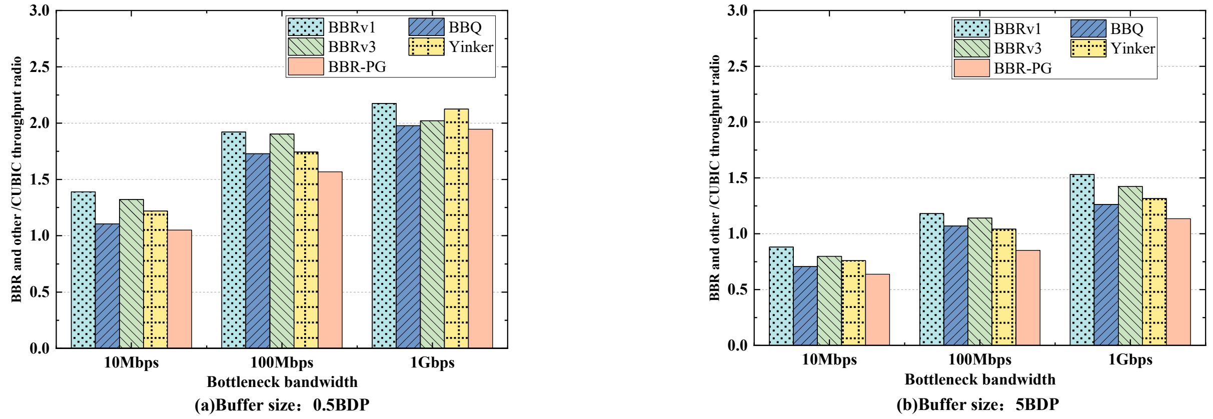
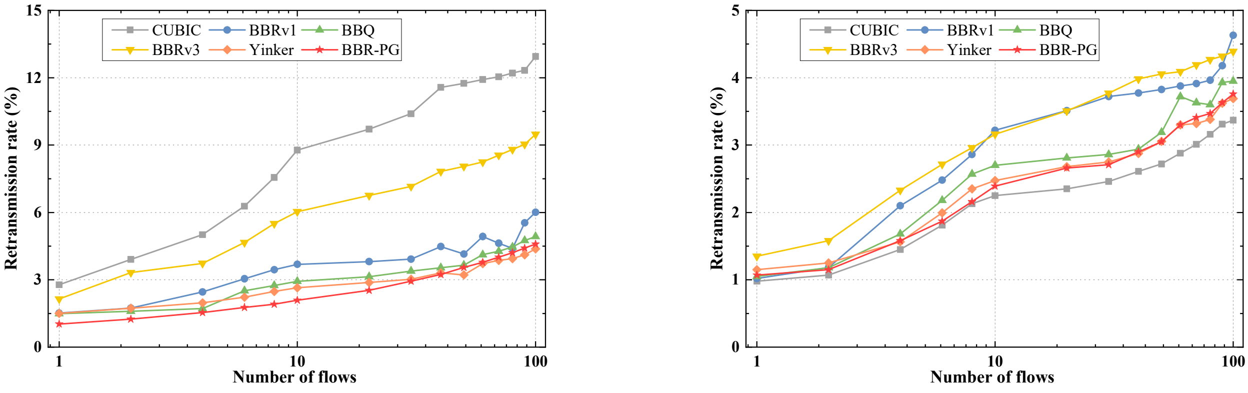


Figure 11. Throughput occupancy ratio of algorithms when coexisting with CUBIC.

4.5. Retransmission

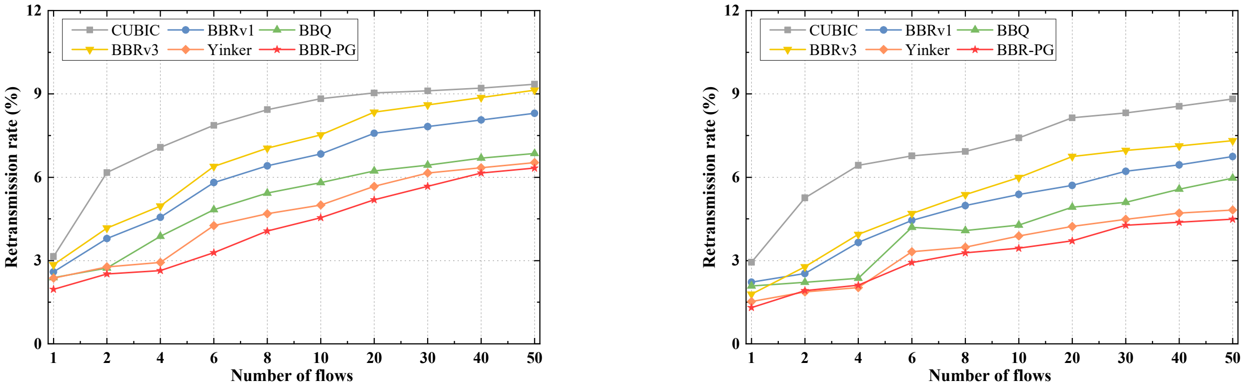
Experiments were performed to verify the effect of different buffer sizes and the number of competing flows on the retransmission rate. The sender used different algorithms to send single or multiple flows to the receiver with buffer sizes of 0.1 BDP or 1 BDP, starting at the retransmission rate of a single 10 ms RTT flow. As shown in Figure 12, the CCAs all experience a large number of retransmissions as the flow number increases. For the small buffer (Figure 12a), the retransmission of CUBIC is significantly higher than that of the other five CCAs. For BBR and its variants, BBRv3 appears to have a high retransmission rate. The retransmission rate of BBR–PG is much lower than that of BBRv1 and BBQ, and slightly lower than that of Yinker by 0.5%. In Figure 12a, the retransmission rates of BBR and its variants are higher than CUBIC's, which can be attributed to the fact that the aggressive de-detection bandwidth due to the step pacing in BBR leads to a certain amount of retransmissions. This needs to be further optimized in future work. However, BBR–PG still achieves the lowest retransmission rate among several BBR variants.



(a) Buffer size：0.1BDP (b) Buffer size：1BDP

Figure 12. Retransmission rates of different numbers of flows in NS3.

We then tested the retransmission rate of the CCAs in 5G and WiFi networks at a buffer size of 0.1 BDP. As shown in Figure 13, the overall retransmission rate in the 5G network environment is slightly higher than that in the WiFi environment. The retransmission rate of CUBIC is significantly higher than that of BBR and its variants, with BBR–PG maintaining the lowest retransmission rate. Similar to the simulation results (Figure 12), the retransmission rate of the CCAs increases as the number of flows increases. However, compared to the simulation results, it can be observed that the retransmission rates of BBRv1 and BBRv3 are significantly higher than those of BBQ, Yinker, and BBR–PG. This indicates that pairing BBRv1 and BBRv3 with a small buffer improves throughput at the cost of a high packet retransmission rate. If the content being transmitted is sensitive to packet loss, then BBRv1 and BBRv3 may not be a good choice. In this case, users must make a trade-off between throughput and quality of experience. In future algorithmic optimization, we can introduce reinforcement learning techniques to accurately predict the network load [31,32]. Based on these forward-looking predictions, the sender can flexibly take appropriate actions, such as adjusting the allocation of network resources and optimizing routing policies, to achieve significant improvements in network performance and stability. Overall, the retransmission rate of BBR–PG is much lower than that of BBRv1 and BBRv3. Experiments have shown that BBR–PG improves data transmission performance in 5G and WiFi networks.



(a) 5G network

(b) WiFi network

Figure 13. Retransmission rates of different numbers of flows in 5G and WiFi networks.

5. Conclusions

In this paper, we have proposed a BBR–PG algorithm to achieve high fairness and low retransmission in high-speed wireless networks. We analyzed and exposed the critical reasons for the fairness problems of BBR. We established the pacing rate control model to analyze the change in actual pacing gain and found that RTT measurement led to the inaccurate evaluation of *BtlBw*. According to the pacing gain mechanism, a modified BBR–PG is proposed that adaptively adjusts the pacing rate by increasing the regulatory factor γ instead of employing the fixed rates of 1.25 or 0.75. Meanwhile, the phase time of the ProbeRTT phase is randomized.

Different network test results show that BBR–PG can effectively alleviate the fairness problem in BBR and retain the advantage of BBR in throughput. In terms of RTT fairness, BBR–PG had the best fairness index. In the NS3 simulation network, its fairness index is 50% higher than BBR, and the RTT fairness of 5G and WiFi networks is also improved. In the inter-protocol fairness tests, the fairness index of BBR–PG in the NS3 simulation network improves by more than 46% compared with BBRv1. The 5G and WiFi network test results also verify that BBR–PG can improve inter-protocol fairness. In addition, BBR–PG can effectively reduce BBR retransmissions. Overall, BBR–PG not only alleviates the fairness problem of BBR but also reduces retransmissions.

In future work, we will continue to optimize the parameters of the BBR–PG algorithm, such as $T_{interval}$ and T_{rand}. We plan to use reinforcement learning to predict network load so that $T_{interval}$ or T_{rand} can be adjusted adaptively.

Author Contributions: Conceptualization, J.X.; methodology, W.P. and J.X.; formal analysis, J.X., W.P. and H.T.; writing—original draft preparation, J.X.; writing—review and editing, X.L. (Xiru Li), H.T., W.P. and J.X.; software, J.X., L.C. and W.P.; visualization, L.C. and X.L. (Xiru Li); supervision, X.L. (Xiaofeng Li). All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Cardwell, N.; Cheng, Y.; Gunn, C.S.; Yeganeh, S.H.; Jacobson, V. BBR: Congestion-Based Congestion Control: Measuring bottleneck bandwidth and round-trip propagation time. *Queue* **2016**, *14*, 20–53. [CrossRef]
2. Chaccour, C.; Soorki, M.N.; Saad, W.; Bennis, M.; Popovski, P.; Debbah, M. Seven defining features of terahertz (THz) wireless systems: A fellowship of communication and sensing. *IEEE Commun. Surv. Tutor.* **2022**, *24*, 967–993. Available online: https://ieeexplore.ieee.org/abstract/document/9681870 (accessed on 2 October 2024). [CrossRef]

3. Lim, H.; Lee, J.; Lee, J.; Sathyanarayana, S.D.; Kim, J.; Nguyen, A.; Kim, K.T.; Im, Y.; Chiang, M.; Grunwald, D.; et al. An empirical study of 5G: Effect of edge on transport protocol and application performance. *IEEE Trans. Mob. Comput.* **2023**, *23*, 3172–3186. Available online: https://ieeexplore.ieee.org/abstract/document/10124095 (accessed on 2 October 2024). [CrossRef]
4. Hock, M.; Bless, R.; Zitterbart, M. Experimental evaluation of BBR congestion control. In Proceedings of the 2017 IEEE 25th International Conference on Network Protocols (ICNP), Toronto, ON, Canada, 10–13 October 2017; pp. 1–10. [CrossRef]
5. Scholz, D.; Jaeger, B.; Schwaighofer, L.; Raumer, D.; Geyer, F.; Carle, G. Towards a deeper understanding of TCP BBR congestion control. In Proceedings of the 2018 IFIP Networking Conference (IFIP Networking) and Workshops, Zurich, Switzerland, 14–16 May 2018; pp. 1–9.
6. Scherrer, S.; Legner, M.; Perrig, A.; Schmid, S. Model-based insights on the performance, fairness, and stability of BBR. In Proceedings of the 22nd ACM Internet Measurement Conference, Madrid, Spain, 25–27 October 2022; pp. 519–537.
7. Cardwell, N.; Cheng, Y.; Yeganeh, S.H.; Swett, I.; Vasiliev, V.; Jha, P.; Seung, Y.; Mathis, M.; Jacobson, V. BBRv2: A model-based congestion control. In Proceedings of the IETF 104th Meeting, Internet Engineering Task Force, Prague, Czech Republic, 23–29 March 2019.
8. Cardwell, N.; Cheng, Y.; Yang, K.; Morley, D.; Yeganeh, S.H.; Jha, P.; Seung, Y.; Jacobson, V.; Swett, I.; Wu, B.; et al. BBRv3: Algorithm Bug Fixes and Public Internet Deployment. In Proceedings of the IETF-117. Internet Engineering Task Force, San Francisco, CA, USA, 26 July 2023.
9. Kfoury, E.F.; Gomez, J.; Crichigno, J.; Bou-Harb, E. An emulation-based evaluation of TCP BBRv2 alpha for wired broadband. *Comput. Commun.* **2020**, *161*, 212–224. [CrossRef]
10. Song, Y.J.; Kim, G.H.; Mahmud, I.; Seo, W.K.; Cho, Y.Z. Understanding of BBRv2: Evaluation and Comparison with BBRv1 Congestion Control Algorithm. *IEEE Access* **2021**, *9*, 37131–37145. Available online: https://ieeexplore.ieee.org/document/9361674 (accessed on 2 October 2024). [CrossRef]
11. Nandagiri, A.; Tahiliani, M.P.; Misra, V.; Ramakrishnan, K.K. BBRvl vs. BBRv2: Examining performance differences through experimental evaluation. In Proceedings of the 2020 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN), Orlando, FL, USA, 13–15 July 2020; pp. 1–6.
12. Drucker, R.; Baraskar, G.; Balasubramanian, A.; Gandhi, A. BBR vs. BBRv2: A Performance Evaluation. In Proceedings of the 16th International Conference on COMmunication Systems & NETworkS (COMSNETS), Bengaluru, India, 3–7 January 2024; pp. 379–387.
13. Zeynali, D.; Weyulu, E.N.; Fathalli, S.; Chandrasekaran, B.; Feldmann, A. Promises and Potential of BBRv3. In Proceedings of the Passive and Active Measurement: 25th International Conference, PAM 2024, Virtual Event, 11–13 March 2024; Proceedings, Part II; Springer Nature: Cham, Switzerland, 2024; pp. 249–272. Available online: https://link.springer.com/chapter/10.1007/978-3-031-56252-5_12 (accessed on 2 October 2024).
14. Hurtig, P.; Haile, H.; Grinnemo, K.J.; Brunstrom, A.; Atxutegi, E.; Liberal, F.; Arvidsson, A. Impact of TCP BBR on CUBIC traffic: A mixed workload evaluation. In Proceedings of the 30th International Teletraffic Congress (ITC 30), Vienna, Austria, 3–7 September 2018; pp. 218–226. [CrossRef]
15. Zhang, Y.; Cui, L.; Tso, F.P. Modest BBR: Enabling better fairness for BBR congestion control. In Proceedings of the IEEE Symposium on Computers and Communications (ISCC), Natal, Brazil, 25–28 June 2018; pp. 00646–00651. [CrossRef]
16. Ware, R.; Mukerjee, M.K.; Seshan, S.; Sherry, J. Modeling bbr's interactions with loss-based congestion control. In Proceedings of the Internet Measurement Conference, Amsterdam, The Netherlands, 21–23 October 2019; pp. 137–143. [CrossRef]
17. Song, Y.J.; Kim, G.H.; Cho, Y.Z. BBR-CWS: Improving the inter-protocol fairness of BBR. *Electronics* **2020**, *9*, 862. [CrossRef]
18. Ishikura, S.; Yamamoto, M. BAR: BBR with Adjusting RTprop for Inter-Protocol Fairness with CUBIC TCP. In Proceedings of the IEEE 29th International Symposium on Local and Metropolitan Area Networks (LANMAN), London, UK, 10–11 July 2023; pp. 1–6.
19. Ma, S.; Jiang, J.; Wang, W.; Li, B. Fairness of congestion-based congestion control: Experimental evaluation and analysis. *arXiv* **2017**, arXiv:1706.09115. [CrossRef]
20. Yang, M.; Yang, P.; Wen, C.; Liu, Q.; Luo, J.; Yu, L. Adaptive-BBR: Fine-grained congestion control with improved fairness and low latency. In Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC), Marrakesh, Morocco, 15–18 April 2019; pp. 1–6. [CrossRef]
21. Kim, G.H.; Cho, Y.Z. Delay-aware BBR congestion control algorithm for RTT fairness improvement. *IEEE Access* **2019**, *8*, 4099–4109. Available online: https://ieeexplore.ieee.org/abstract/document/8943219 (accessed on 2 October 2024). [CrossRef]
22. Pan, W.S.; Li, X.F.; Tan, H.B.; Xu, J.L.; Li, X. Improvement of RTT Fairness Problem in BBR Congestion Control Algorithm by Gamma Correction. *Sensors* **2021**, *21*, 4128. [CrossRef] [PubMed]
23. Njogu, C.K.; Yang, W.; Njogu, H.W.; Bosire, A. BBR-With Enhanced Fairness (BBR-EFRA): A new enhanced RTT fairness for BBR congestion control algorithm. *Comput. Commun.* **2023**, *200*, 95–103. [CrossRef]
24. Kanaya, T.; Tabata, N.; Yamaguchi, S. A study on performance of CUBIC TCP and TCP BBR in 5G environment. In Proceedings of the IEEE 3rd 5G World Forum (5GWF), Bangalore, India, 10–12 September 2020; pp. 508–513.
25. Sandoval, J.I.; Céspedes, S. Performance evaluation of congestion control over b5g/6g fluctuating scenarios. In Proceedings of the Int'l ACM Symposium on Design and Analysis of Intelligent Vehicular Networks and Applications, Montreal, QC, Canada, 30 October–3 November 2023; pp. 85–92.

26. Lee, C.; Higuchi, T.; Ucar, S.; Kaneko, N.; Altintas, O.; Oguchi, K. Poster: Performance Analysis of TCP CUBIC and BBR over V2V Wi-Fi. In Proceedings of the 22nd Annual International Conference on Mobile Systems, Applications and Services, Minato-ku, Tokyo, Japan, 3–7 June 2024; pp. 668–669.
27. Guo, L.; Liu, Y.; Yang, W.; Zhang, Y.; Lee, J.Y. Stateful-bbr–an enhanced tcp for emerging high-bandwidth mobile networks. In Proceedings of the IEEE/ACM 29th International Symposium on Quality of Service (IWQOS), Tokyo, Japan, 25–28 June 2021; pp. 1–9.
28. Xie, Y.; Jiang, X.; Gong, G.; Jiang, Z.; Jin, G.; Chen, H. Yinker: A flexible BBR to achieve the high-throughput and low-latency data transmission over Wi-Fi and 5G networks. *Comput. Netw.* **2023**, *222*, 109530. [CrossRef]
29. Ahsan, M.; Muhammad, S.S. TCP BBR-n: Increased throughput for wireless-AC networks. *PLoS ONE* **2023**, *18*, e0295576. [CrossRef]
30. Jain, R.K.; Chiu, D.M.W.; Hawe, W.R. *A Quantitative Measure of Fairness and Discrimination*; Eastern Research Laboratory, Digital Equipment Corporation: Hudson, MA, USA, 1984; p. 21. Available online: https://ocw.cs.pub.ro/courses/_media/isrm/laboratoare/new/a_quantitative_measure_of_fairness_and_d.pdf (accessed on 2 October 2024).
31. Alawe, I.; Ksentini, A.; Hadjadj-Aoul, Y.; Bertin, P. Improving traffic forecasting for 5G core network scalability: A machine learning approach. *IEEE Netw.* **2018**, *32*, 42–49. Available online: https://ieeexplore.ieee.org/document/8553653 (accessed on 2 October 2024). [CrossRef]
32. Spantideas, S.; Giannopoulos, A.; Cambeiro, M.A.; Trullols-Cruces, O.; Atxutegi, E.; Trakadas, P. Intelligent Mission Critical Services over Beyond 5G Networks: Control Loop and Proactive Overload Detection. In Proceedings of the 2023 International Conference on Smart Applications, Communications and Networking (SmartNets), Istanbul, Turkey, 25–27 July 2023; pp. 1–6.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.